VMSAGE: A Virtual Machine Scheduling Algorithm based on the Gravitational Effect for Green Cloud Computing

Xiaolong Xu[a], Qitong Zhang[b], Stathis Maneas[c], Stelios Sotiriadis[d,*], Collette Gavan[e], Nik Bessis[e]

[a]*School of Computer Science, Nanjing University of Posts & Telecommunications, Nanjing 210003, China*
[b]*State Key Laboratory of Information Security, Chinese Academy of Sciences, Beijing 100093, China*
[c]*Department of Computer Science, University of Toronto, Canada*
[d]*Department of Computer Science and Information Systems, Birkbeck, University of London, Malet St, Bloomsbury, London WC1E 7HX, UK*
[e]*Department of Computer Science, Edge Hill University, Ormskirk, Lancashire L39 4QP, UK*

Abstract

The area of sustainable green smart computing highlights key challenges towards reducing cost and carbon dioxide emissions due to the high-energy consumption of Cloud data centres. Here, we focus on the Cloud virtual machine (VM) scheduling that is usually based on simple algorithms, e.g. VM placement on nodes with low memory usage. This approach fails to consider the actual configuration of nodes inside the server rack resulting in local overheating of Cloud data centres. To solve this, we propose a VM scheduling algorithm based on the gravitational effect, called *VMSAGE*, to optimize energy efficiency of Cloud computing systems. Inspired by the physical gravitation model, we define the thermal repulsion and logical gravitation factors between physical nodes and VMs. To achieve optimized VM scheduling, we propose a gravitation function that refers to the calculation of the logical quality of each VM, host and rack through the algorithm, so as to draw the attractiveness between

*Corresponding author
Email addresses: xuxl@njupt.edu.cn (Xiaolong Xu), 1014041119@njupt.edu.cn (Qitong Zhang), smaneas@cs.toronto.edu (Stathis Maneas), stelios@dcs.bbk.ac.uk (Stelios Sotiriadis), Gavanc@edgehill.ac.uk (Collette Gavan), Nik.Bessis@edgehill.ac.uk (Nik Bessis)

them. Based on the concept of dimension reduction, *VMSAGE* conducts the two-dimensional plane target selection twice to reduce the computational cost. Additionally, *VMSAGE* evaluates attributes of the computer room to carry out the VM deployment. To demonstrate the effectiveness of our solution, we compare it with the Best Fit Heuristic (BFH) and the dynamic voltage and frequency scaling (DVFS) algorithms. The results indicate that our algorithm achieves 10% and 20% optimized energy consumption respectively. The experimental results highlight our contribution, in where *VMSAGE* can significantly reduce energy consumption rates and VM migration times.

Keywords: Virtual Machine, Scheduling Algorithm, Energy Efficiency, Gravitation Effect, Cloud Computing

Funding: This work was jointly sponsored by the National Natural Science Foundation of China under Grant 61472192 and 61472193, and the Natural Science Foundation of Jiangsu Province under Grant BK20141429.

---

1. Introduction

Cloud computing enables an economically promising paradigm of computation outsourcing [1]. Immense computation power and storage capacity of computing systems enable everyday Internet users to store and process large-scale data on a "pay as you go" model [2]. As more users move their activities to the Cloud, the number of data centre nodes increases as well. The global data center market is estimated to reach revenues of around 174 billion by 2023, growing at a Compound Annual Growth Rate (CAGR) of approximately 4% during the forecast period[1].

The demand for data centres processing capacity is expected to increase by 7 to 10 times in the next 5 years [3]. However, as the scale of Cloud data centres increases, the physical servers cause high power consumption and environ-

[1] https://www.prnewswire.com/news-releases/data-centers-global-industry-outlook–forecast-2018-2023—adoption-of-hyperconverged-infrastructure-to-have-a-high-impact-on-growth-300676458.html

ment problems. Today, the annual power consumption of global data centres is about 3,000 tw.h, equivalent to the total power generation of 300 nuclear power plants [4]. For example, the annual power consumption of Google's Cloud data centres is up to nearly 203,000,000 kw.h [4]. The inefficient utilization of resources causes unnecessary waste of energy. Kurnik et al. [35] show that the current under-utilization rates of many servers in data centres are around 90%.

The effective virtualization of resources can be used to solve the low energy efficiency problem. In Cloud systems, nodes are virtualized in a unified and on-demand resource pool, which improves the utilization rates of resources and reduces the required number of computing nodes to a certain extent [6]. De Assuncao et al. [7] suggest that during an infrastructure's high energy consumption period, for each 1 kw.h power used for computing, there is an additional 0.5-1 kw.h power used for cooling. A physical node with low utilization rate of resources can switch to a sleeping state, or even shut down by migrating its VMs dynamically to other data nodes, in order to attain the better energy-saving effect. In order to effectively improve the service performance of Cloud computing systems, while at the same time also reduce the cost of service and energy consumption, we suggest that efficient VM monitoring management, scheduling and migration algorithms and strategies are essential.

Many research results have been proposed in this field, such as the migration cost-aware locally optimal placement algorithm (pMaP) [8], the peak clustering-based placement (PCP) [8], the minimum migration time cuckoo optimization algorithm (COA-MMT) [9], the minimum migration time imperialism competitive algorithm (ICAMMT) [10], etc. However, three major problem-specific challenges make the solution a complex task:

1. The virtual machine scheduling and migration are only conducted on data, such as CPU utilization and RAM remaining space. Due to the random distribution of active nodes in Cloud data centres, it is difficult to implement regional precise temperature control.

2. VM scheduling and migration requirements are considered logically, while

the actual data node deployment scenarios, the heat distribution, and the operation mode of the temperature control system of Cloud data centre are not. Due to the poor linkage between data nodes and the temperature control equipment, it is difficult for these algorithms to be applied in actual Cloud data centres.

3. Data center overheating is problematic, which makes it difficult to solve, compromising the maintenance of the system's stability. The local overheating problem of data centre affects the lifetime of computing, storing and communication equipment. The throughput and latency in future data centre networks must be significantly improved to sustain the increased network traffic and the total power consumption inside the racks must remain almost the same due to thermal constraints [11]. If a node's dataset involves heavy computing tasks with relatively poor thermal performance, it might lead to abnormal behaviour (e.g. shut down due to local overheating) [12].

In order to achieve the goals of load balance, energy efficiency, service-level agreement (SLA) and stability, Cloud data centres need more reasonable algorithms for scheduling, migration and management of VMs. Considering the actual server deployment and the temperature control mode of real Cloud data centres, we propose a novel VM scheduling algorithm based on the gravitational effect (*VMSAGE*).

The contribution of this work is that *VMSAGE* creatively utilizes the concept of physical gravitational effect and defines various attributes, including the thermal repulsion factor between the physical node and a virtual machine, the logical gravitation and the modified gravitation, along with their calculation functions. Specifically, our approach is based on the following three novel elements:

(a) **Thermal Repulsion Factor**. The thermal repulsion factor not only does it achieve the diffusion of the VMs from overheated data nodes in order to stabilize both the entire system and the local systems, but also prevents

data nodes from collapsing and from further damage. This guarantees quality of service (QoS) while keeping SLAs.

(b) **Logical Gravitation.** The logical gravitation aggregates VMs to specific data nodes. We shut down those data nodes when a low resource utilization rate is observed.

(c) **Accord Priority to Heat Dissipation.** Accord priority to heat dissipation, the modified gravitation analyzes the heat distribution in Cloud data centres and makes the virtual machines migrate to the data nodes with good heat dissipation performance to achieve the balance of heat distribution and avoid overheated server racks.

In Section 2, we describe previous work in the area. Section 3 describes the scheduling algorithm, while Section 4 contains the implementation details of VMSAGE. In Section 5, we conduct our performance evaluation and finally, in Section 6, we conclude.

2. Related Work

This section presents the related work in the area of data centre temperature control and VM scheduling.

The data centre as a Computer, described by Google [13] treats the data centre as a multi-function building, capable of accommodating multiple servers and communication equipment. Rather than having a collection of servers, these devices are placed together as they have the same environmental and security requirements making maintenance more efficient.

Inside a data centre, the most quantity of heat is generated by clusters of servers, accounting for about 60% to 70% of the total heat quantity. In order to reduce the energy consumption of a data centre, the machines with better ability of dissipating the heat should be preferred when it comes to provide services.

Generally, the temperature control equipment used by a data center is either air cooling or liquid mode, with the underfloor air supply system being the most

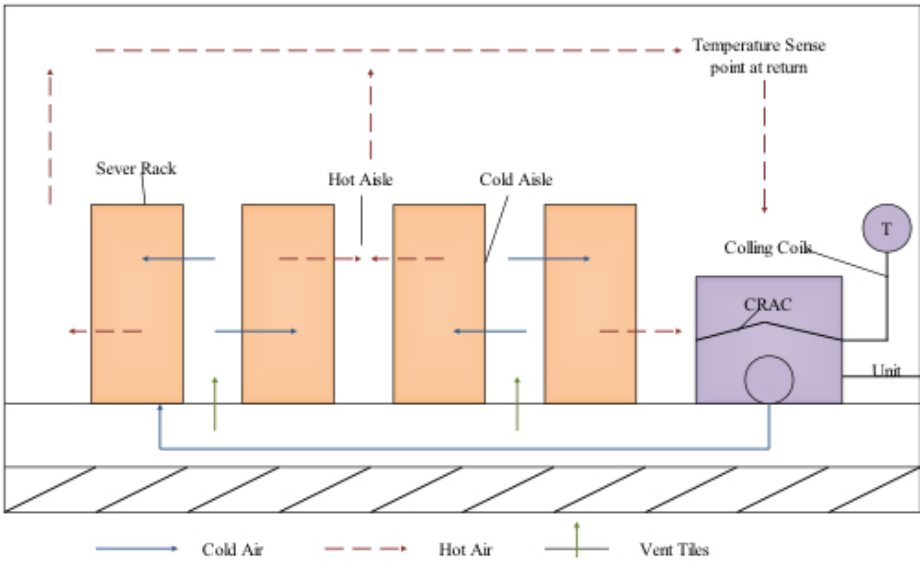


Figure 1: *Schematic diagram of Cloud data centre based on air cooling temperature control. The Computer Room Air Conditioner (CRAC) is a device that monitors and controls the temperature, air distribution, and humidity in a network room or data center, while T denotes the temperature sensor.*

popular, as shown in Figure 1. The shortcomings of this temperature control mode are the following:

1. Workloads of servers, which are temporally and spatially uneven, resulting in chaotic distribution of hot spots and cold spots in data center are ignored.

2. The temperature control module works on a thermodynamic stable system, which leads to less than 50% of effective cooling capacity.

3. The servers on the top of server racks can easily fail due to overheating [10].

The efficient migration of VMs needs both an efficient scheduling algorithm, along with a dynamic migration copy algorithm. The typical dynamic migration copy modes of VMs include the Pre-Copy, Post-Copy and CR/RT-Motion algorithms [14]. However, these algorithms are out of this paper's scope and therefore, they will not be discussed in further detail. Instead, we will focus on

VM scheduling algorithms.

In order to reduce energy consumption, Zhao et al. [8] propose a virtual machine scheduling algorithm based on energy minimization. According to the energy consumption model, the algorithm calculates the utilization rate of data nodes using a greedy method and repeatedly selects the data node with the least growth of energy consumption, as the operation location for VMs, until all the locations for VMs are determined. Then, the virtual machine deployment task is completed based on the first fit decreasing algorithm. By deploying virtual machines on a predetermined set of nodes, the algorithm can minimize the data nodes' energy consumption and achieve the ideal utilization rate of each data node at the same time. Chen et al. [15] try to mitigate the impact of uncertainty on the task scheduling quality for Cloud data centres. They present a novel scheduling algorithm, which dynamically exploits the proactive and reactive scheduling methods for scheduling real-time, aperiodic and independent tasks. To improve the energy efficiency, they further propose strategies to scale up and down the system's computing resources based on the executed workloads, resulting in improved resource utilization rates and reduced energy consumption inside a Cloud data centre. Beloglazov et al. [16] present an analytical study to explore single VM migration and consolidation, along with their respective on-line deterministic algorithms. The authors provide adaptive heuristics (VM placement optimization and power aware best fit decreasing) that analyze historical data of servers with regards to resource usage for optimizing energy and performance of VMs. The aforementioned solutions [8, 15, 16] though do not focus on the physical structure of a data centre and also fail to consider the problems provoked by local overheating, long downtime of servers and unsatisfying energy conservation.

Yakhchi et al. [9] propose a load balancing algorithm called COA-MMT to reduce the energy consumption of Cloud computing infrastructures. Compared with other classical algorithms, such as Interquartile Range-Minimum Migration Time (IQR-MMT) and Median Absolute Deviation-Minimum Migration Time (MAD-MMT), the efficiency of a load balancing method based on Cuckoo

Optimization Algorithm for Energy Management (COA-MMT) is significantly improved from the aspect of VM migration, although it does not consider the response time factor of QoS. Zhu et al. [17] present their work to improve the ability of migrating VMs inside a data centre to help improving the capacities of dynamic resource load balancing and fault tolerance. The migration of VMs includes the migrations of their memory state, application program and operating system.

Fang et al. [18] aim to optimize VM placement and traffic flow routing by presenting them as optimization problems. The authors suggest that $10 - 20\%$ of the total power consumption is provoked by the network elements. Therefore, they propose "VMPlanner", which includes three approximation algorithms: (a) VM grouping according to minimized traffic volume, (b) VM-group to server-rack mapping (for placing VMs into rack more efficiently), and (c) power-aware inter-VM traffic flow routing for minimizing the number of paths in the network. They use the Greedy Bin-Packing algorithm, as presented in [19] to select the path with the most sufficient capacity.

In [20], the authors study load placement policies for cooling and data centre temperatures. They follow the idea of VM placement and migration according to different electricity prices and temperatures during the time; basing their system on the assumptions of users who provided estimates of the running times of jobs. They use a round-robin algorithm and a cost aware static policy for comparing their results.

Tseng et al. [21] explore the service-oriented VM placement as an optimization problem. The authors try to solve the problem using a graph of Tree (to minimize communication between VMs) and Forest algorithm (for balancing traffic load between VMs). To evaluate the algorithm, they present a comparison against the Best Fit algorithm. They claim that the Forest algorithm decreases the outbound communication cost by 22% and the Tree algorithm by 92%. Cardosa et al. [22] present their approach on VM placement and consolidation of VMs in Cloud data centres using min-max. The authors suggest that guided by application utilities, they could provide better resource allocation, so

that high utility applications get most of the resources. They experiment using synthetic and real data centre test-beds, concluding that the *PowerExpandMinMax* algorithm is the best utility for power-performance trade-offs in modern data centres running heterogeneous applications.

In [23], the authors present a VM placement framework for minimizing the energy consumption and maximizing the profits. They treat these as constraint satisfaction problems and propose a utility function that expresses the SLA satisfaction. From the system's perspective, the authors provide a VM placement formulation in order to maximize the number of the machines that must be turned off. The experimental analysis is based on the Xen hypervisor and the results indicate that different parameters can affect the operational costs and can be used to balance the QoS.

In [24], the authors present a resource allocation problem that aims to minimize the total energy cost of a Cloud system in a probabilistic way. Their algorithm places virtual machines (VMs) to servers using dynamic programming and convex optimization. The algorithm is evaluated using simulations and the results show that the proposed VM placement minimizes the power cost. Goudarzi et al. [25] try to solve the Cloud energy-efficient VM placement by creating multiple copies of VMs and by placing them into PMs using local search. The experimental analysis shows a 20%-improvement in energy consumption compared with selected heuristics.

In [26], the authors provide a framework, which allows the allocation of VMs in a data centre in such a way to achieve energy awareness. The authors further decouple constraints from algorithms by implementing 16 frequently used SLA parameters in the form of constraints. The experimental analysis shows an 18%-improvement in savings of both energy and CO_2 emissions. EnaCloud [27] is an energy aware heuristic algorithm for VM placement in a dynamic way, also considering energy efficiency. The authors present an experimental case for the Xen VMM, claiming that their solution saves energy in Cloud platforms by comparing their approach with FirstFit and BestFit. Finally, they separate their workloads into different types, such as web/database server, compute-intensive

and common applications.

Regarding VM migration time, Baruchi et al. [28] define the work cycle of a VM, and VM migration after a work cycle, to effectively reduce the network traffic and migration time. In [29], the authors present their study on how to migrate virtual machines to appropriate servers, in order to reduce the number of VM migrations. Their study focuses on 3 issues which are important to the efficiency of VM migration, namely how to allocate VMs, how to migrate VMs from heavily loaded servers, and how to select the right VMs to migrate.

Horvath et al. [33] explore the benefits of dynamic voltage scaling (DVS) for power management in server farms. The authors present a rigorous optimization methodology and an algorithm for minimizing the total energy expenditure. They design a distributed power management prototype for coordinating DVS settings in such a way to minimize the global energy consumption. Mofolo et al. [34] propose a Best Fit Heuristic (BFH) algorithm, along with a Best Fit Decreasing (BFD) algorithm that improves the speed of host selection and VM reallocation. The BFH algorithm can be used to achieve maintenance of physical servers and can lead to efficient consolidation of VMs.

Our study on the existing literature indicates that the existing approaches implement deployment and migration of virtual machines based on CPU utilization and remaining memory (RAM) of data nodes, without really integrating other resources in the global scope of Cloud data centres. The discreteness of active nodes makes it difficult to implement precise temperature control. Furthermore, these approaches do not consider the node deployment, the heat distribution, and the relation between computing devices and temperature control devices. In our work, we propose a new strategy of virtual machine management by combining the physical concepts with the actual situation of Cloud data centres.

3. VM Scheduling Algorithm based on the Gravitational Effect

We begin this section by introducing the necessary definitions for our algorithm's operation and then, we present our VM scheduling algorithm based on the gravitational effect. Our algorithm aims at providing the following:

1. **Temperature balance:** We use orderly aggregation of VMs to avoid local overheating caused by excessive computing tasks or poor heat dispersion performance of nodes.

2. **Energy consumption reduction:** Certain computing nodes and the corresponding cooling devices can be shut down through the aggregation of VMs to save as much energy as possible without affecting the performance of VMs.

3. **Operation stability:** The bumpiness, defined as the phenomenon where a VM needs to migrate again or migrate back to the server from which it just migrated away, can be prevented, so that QoS can be ensured.

Parameter	Definition
f_t	All the racks in the data centre are numbered.
	The t-th rack is defined as f_t.
s_{ti}	All the servers in the data centre are numbered.
	The i-th server on the t-th rack is defined as s_{ti}.
ν_{ij}	All the VMs in the data centre are numbered.
	The j-th VM on the i-th server is defined as ν_{ij}.
r_m	All the routers in the data centre are numbered.
	The m-th router is defined as r_m.
h_n	All the switches in the data centre are numbered.
	The n-th switch is defined as h_n.
t_i	The temperature of the i-th server is defined as t_i.

Table 1: *Related Basic Parameters of* VMSAGE.

The main idea of the gravitation model is based on the work of [30], which includes a function to describe spatial interaction. We extend this model since the current theoretical basis of this gravitation model leads to limitations in its practical applications. In this work, we suggest that by transforming the gravitation model from an empirical to a theoretical one, we can apply the concept of gravitation to a variety of logical objects. In order to achieve the aforementioned goals, we propose *VMSAGE*, the VM scheduling algorithm based on the gravitational effect for Cloud computing infrastructures. Our configuration parameters are i) the thermal repulsion factor (for the upper limit of temperature) and ii) the logical quality and distance (parts of the gravitation calculation). Table 1 presents the definition of the parameters involved in *VMSAGE*. We proceed by providing the necessary *VMSAGE* definitions.

Definition 1: Logical quality. Logical quality $M(x)$ represents the amount of resources an object x occupies inside the data centre. For example, the logical quality of VM ν_{ij} is defined as $M(\nu_{ij})$, the logical quality of server s_{ti} is defined as $M(s_{ti})$, and the logical quality of server rack f_i is defined as $M(f_i)$:

$$M(\nu_{ij}) = \sum_{l \in J} k_l \times U_l(\nu_{ij}), J \in \{CPU, RAM, DISK, I/O, ...\} \quad (1)$$

$$M(s_{ti}) = \sum_{l \in J} k_l - \sum_{\forall \nu_{ij}} M(\nu_{ij}) - \alpha \times T(t_i), J \in \{CPU, RAM, DISK, I/O, ...\} \quad (2)$$

$$M(f_t) = \sum_{\forall s_{ti}} M(s_{ti}) \quad (3)$$

Equation (1) calculates the sum of the resources occupied by ν_{ij}, where k_l is the empirical coefficient and U_l is the resource sharing factor of ν_{ij} related to s_{ti}. In general, if a particular service of a server is intensive, the corresponding empirical coefficient is big. In Equation (1), the numerical values of resource utilization are in different units and can be unified to obtain a uniform evaluation function to get the logical quality of a virtual machine.

For example, suppose that there is a virtual machine, called ν_{11}, located on server s_{01}, which is CPU-intensive. In this setup, we consider CPU, RAM,

and I/O. The current occupancy of the ν_{11} server regarding CPU, RAM and I/O is 10%, 5%, and 7% respectively. The empirical coefficient for CPU is set relatively high because of our assumption that s_{01} is a CPU-intensive server. Therefore, $J \in \{CPU, RAM, I/O\}$, and $k_1 = 6$, $k_2 = 2$, and $k_3 = 2$. The logical quality of ν_{11} can be calculated as $M(\nu_{11}) = 6 \times 0.1 + 2 \times 0.05 + 2 \times 0.07 = 0.84$.

Equation (2) calculates the logical quality of server s_{ti}, where α denotes the correction coefficient, while $T(t_i)$ denotes the thermal repulsion coefficient used to reflect the temperate state of the server. The equation calculates the logical quality of a server by adjusting its spare resources with $T(t_i)$ and thus, it represents a server's subsequent attraction to VMs. Finally, Equation (3) calculates the sum of idle resources on rack f_t, that is, the logical quality of f_t.

In a real data centre, $T(t_i)$ is calculated based on the following formula,

$$T(t_i) = t_i + \beta(t_i)$$

where t_i can be calculated by the prediction calculation function obtained by evaluating the performance of this server model, while $\beta(t_i)$ is used in order to correct deviations. Both α and $\beta(t_i)$ are correction coefficients that can be adjusted with real-time monitoring of the cloud data centre. They can be determined based on different parameters, such as the air cooling mode and the position of the server inside its rack.

In order to establish the link between power consumption and the temperature of the CPU, we used Matlab to implement the required experiment. The results shown in Figure 2 indicate that a) as the load of the CPU changes (from the non-load state to the medium-load state), the power consumption grows significantly and that b) as the medium-load state changes to the full-load state, the growth of its power consumption is slowing down. Based on the gray prediction method to fit the data, the temperature prediction function suitable for this specific data centre can be designed as:

$$t_i = 28 \times e^{0.0014 M(s_{ti})}$$

In this case, the data centre uses the air cooling mode, which means that the

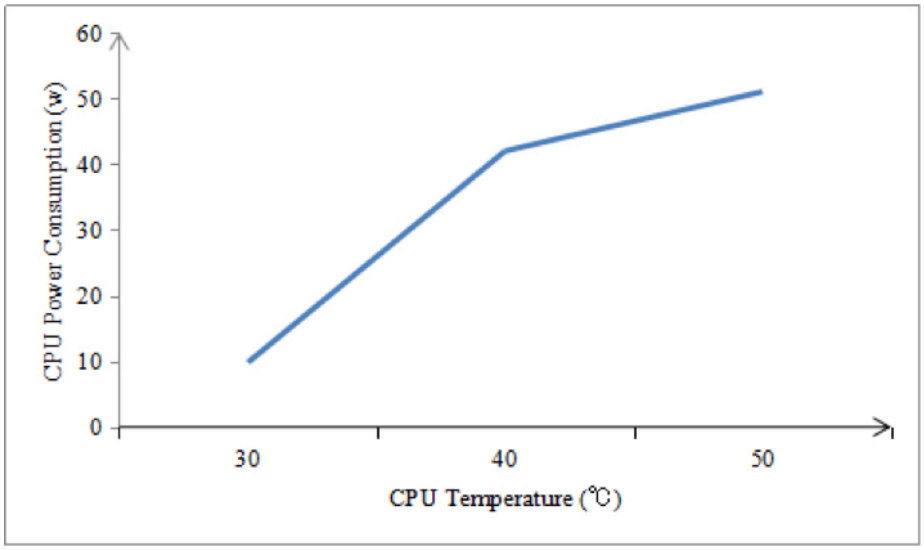


Figure 2: *Link of the power consumption and the temperature of the CPU.*

higher position of the server rack, the poorer performance of the heat dissipation. For each layer that is higher in the rack, the temperature will rise by 2.7^oC to 3.4^oC. In this case, the predictive value of $T(t_i)$ is equal to,

$$T(t_i) = 28 \times e^{0.0014M(s_{ti})} + 3 \times l$$

where l is the layer counting from bottom to top. The upper limit thresholds of the utilization rates of resources and the temperature that the server can bear should be set in such a way to ensure that the selected server is available for creating VMs local or accepting VMs migrated from other servers. If the resource utilization rate of the server is too low, VMs should be migrated, so that the server can enter a dormant state to save power. The servers of a data centre are classified into four types, as shown in Table 2.

Since resource utilization and temperature are related to the logical quality, such thresholds can be delimited by the logical quality $M(s_i)$ in a uniform

Type	Definition
A	VMs running on the server should be migrated, and then the server can hibernate.
B	The server can accept more VMs to achieve the high utilization rate of its resources.
C	VMs running on the server do not need to be rescheduled.
D	The temperature or the workload of the server is too high, or its workload is too heavy and some VMs running on the server need to be migrated out the soonest possible.

Table 2: *Server Classification.*

manner. The server classification rule is defined as follows:

$$\begin{cases} \text{if } M(s_{ti}) \leq \gamma_1, & s_{ti} \in A \\ \text{if } \gamma_1 < M(s_{ti}) \leq \gamma_2, & s_{ti} \in B \\ \text{if } \gamma_2 < M(s_{ti}) \leq \gamma_3, & s_{ti} \in C \\ \text{if } \gamma_3 < M(s_{ti}), & s_{ti} \in D \end{cases}$$

where γ_1, γ_2, and γ_3 are threshold values.

Definition 2: Logical distance. Logical distance refers to the data transmission cost between two servers in a data centre. We define $L(x, \nu_{jk})$ as the cost of data transmission when a VM ν_{jk} migrates from server j to server x, which represents the logical distance and can be calculated using the following formula,

$$L(x, \nu_{jk}) = \sum_{0}^{y_1 - 1} \frac{B(\nu_{jk})}{B_F(h_y)} + \sum_{0}^{y_2 - 1} \frac{B(\nu_{jk})}{B_F(r_y)} + 1$$

where k_1 and k_2 are both empirical coefficients, $B(\nu_{jk})$ is the network resource occupied by VM ν_{jk}, $B_F(h_y)$ is the idle bandwidth of switch h_y, $B_F(r_y)$ is the idle bandwidth of router r_y, y_1 is the number of switches VM ν_{jk} passes by, and finally, y_2 is the number of routers VM ν_{jk} passes by.

Definition 3: Logical gravitation. Logical gravitation $\overrightarrow{F}$ refers to a mutual attraction between any two objects. The value of $\overrightarrow{F}$ is directly proportional to the product of their logical quantities and inversely proportional to the square of their logical distance.

The gravitational effect is used here to reflect the ability of aggregating and attracting VMs. An appropriate gravitation model of the data centre must be established. In accordance to [30], we design the following logical gravitation model,

$$\overrightarrow{F} = G\frac{Mm}{r^b}$$

where M and m represent the two objects, while G and b are the constant coefficients. The logical gravitation model can be derived as:

$$\begin{cases} \overrightarrow{F}(f_t, \nu_{jk}) = G\frac{M(f_t)M(\nu_{jk})}{L(f_t,\nu_{jk})^b} \\ \overrightarrow{F}(s_{ti}, \nu_{jk}) = G\frac{M(s_{ti})M(\nu_{jk})}{L(s_{ti},\nu_{jk})^b} \end{cases}$$

Under the control of the logical gravitation, VMs will rapidly aggregate to some data nodes in the data centre. Then, there will be some data nodes which are idle and can hibernate, so that the corresponding temperature control system can also be quickly adjusted. As a result, the data centre can save energy on both computing and temperature control. The resultant force on virtual machine ν_{jk} from x server racks is equal to:

$$\overrightarrow{F}_f(\nu_{jk}) = \sum_{i=0}^{x-1} \overrightarrow{F}(f_t, \nu_{jk})$$

The resultant force on virtual machine ν_{jk} from $y+1$ servers is equal to:

$$\overrightarrow{F}_s(\nu_{jk}) = \sum_{i=0}^{y} \overrightarrow{F}(s_{ti}, \nu_{jk}) \tag{4}$$

Definition 4: Modified gravitation. Modified gravitation refers to the force that attracts VMs toward the direction of a rack with better cooling performance in the data centre due to the temperature effect. The modified gravitation on

ν_{jk} is defined as follows,

$$\overrightarrow{G}(\nu_{jk}) = g \times M(\nu_{jk}) \tag{5}$$

where g is the correction factor.

Logical gravitation is used to reflect the current server load capacity, while the modified gravitation is used to reflect the future server load capacity. Obviously, a server with good heat dissipation performance can prevent downtime and carry more tasks. According to the definitions of logical quality and logical gravitation, other concepts such as logical acceleration and logical speed can also be defined. For example, if a virtual machine is attracted by several server racks, it will automatically move to that sever rack with the biggest logical force (that defines the acceleration of VM migration). Since $\overrightarrow{F}_s(\nu_{jk})$ and $M(\nu_{jk})$ are both known, the acceleration of ν_{jk} migration, denoted by $a(\nu_{jk})$, is calculated using the following formula:

$$\overrightarrow{a}(\nu_{jk}) = \frac{\overrightarrow{F}_s(\nu_{jk})}{M(\nu_{jk})} \tag{6}$$

In the physical world, under the same acceleration, the higher initial velocity the object has, the sooner it will reach the destination. Here, we use the initial speed of VM migration to reflect the priority of VM migration, so that the system can decide the migration order of VMs.

The $\nu_0(\nu_{jk})$ term is used to represent the initial speed associated with the migration related to ν_{jk}, where a reference value of initial velocity ν_0 and a constant β are given. A type D server hosts VMs that have high initial speeds, because they are in urgent need to migrate to other available servers. These VMs are sorted (in descending order) according to the amount of resources they occupy; VMs with low utilization of resources have low initial speeds. After the migration of VMs, the server's type will change to C. The minimum initial speed of the VM is defined as ν_0, and the initial speed of each VM is increased by β successively.

Servers of type A, require hibernation. All servers of type A are sorted in accordance with their current energy consumption in an ascending order.

The initial speeds of all VMs running on the server with the minimum energy consumption are defined as $\nu_0 - \beta$, and the initial speeds of VMs running on other servers are reduced by β successively. In different systems, values of ν_0 and β will be different.

Speed of VM migration. As the system's logic clock p goes on, the migration speed of ν_{jk} (denoted as $V(\nu_{jk})$) is defined as:

$$V(\nu_{jk}) = \nu_0(\nu_{jk}) + \alpha(\nu_{jk}) \times p$$

Distance of VM migration. According to Equations (4), (5), and (6), the migration distance of ν_{jk}, denoted as $X(\nu_{jk})$, is defined as:

$$X(\nu_{jk}) = \alpha(\nu_{jk}) \times p + \frac{1}{2}\alpha(\nu_{jk}) \times p^2$$

The gravitation algorithm of *VMSAGE* needs to calculate the distance between a VM and its target rack and then, determine the VM migration decision. The rack selection workflow of the gravitation algorithm proceeds as follows:

Step1: Get the server lists of type D and type A. Then, calculate the logical quality of each server rack.

Step2: Assign the initial speeds of all the VMs running on all servers of type D and type A. The initial state of a data centre is shown in Figure 3a.

Step3: Calculate the logical gravitation of each server rack to the VM that must be migrated. The direction of gravitation is pointing to the target rack. As shown in Figure 3b, F_1 is the logical gravitation from rack 1 to the VM, F_2 is the logical gravitation from rack 2 to the VM, and finally, F_3 is the logical gravitation from rack 3.

Step 4: Calculate the logical distance of each VM moving. If the logical distance is equal to the distance from this VM to its target rack, the VM will be migrated, as shown in Figure 3b.

Step5: If the VM has not been fully migrated yet, go back to Step 4. Otherwise, the algorithm ends, as shown in Figure 3c.

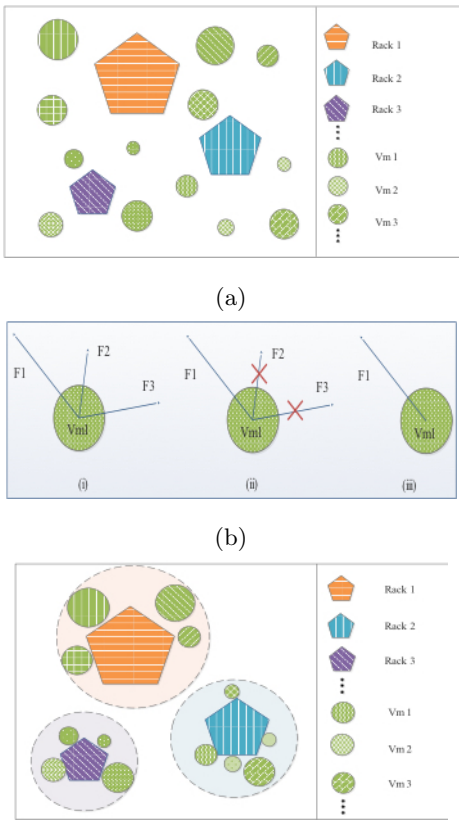


Figure 3: *The rack selection diagram of the gravitation algorithm.*

After the target rack has been selected, we determine the modified gravitation that is used to adjust the migration direction, so that a server with good heat dissipation performance is chosen. The modified gravitation on ν_{jk} is defined as follows:

$$\overrightarrow{F'}_s(\nu_{jk}) = \sum_{i=0}^{y} \overrightarrow{F'}(s_{ti} + \nu_{jk}) + \overrightarrow{G}(\nu_{jk})$$

In accordance with Equation (6), the target server which $\overrightarrow{F'}_s(\nu_{jk})$ points to can be determined. Figure 4a shows a side view of a server rack, where $m \times n$ squares represent $m \times n$ servers. Figure 4b shows the force from a rack to a VM. Its values are calculated at the origin $(0, 0)$, i.e., the middle of the frame.

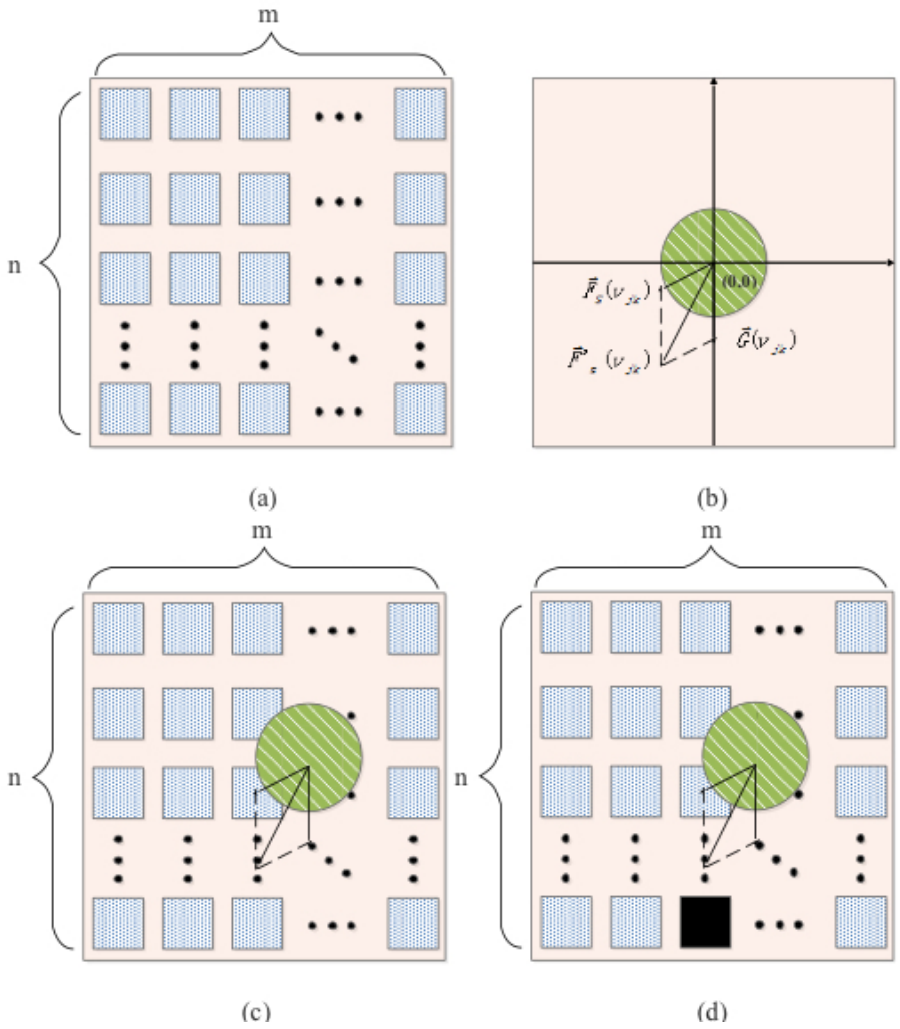


Figure 4: *Diagram of force on a virtual machine.*

Figure 4c and Figure 4d show the selection of the target server based on the resultant force; the black box represents the selected server.

4. Implementation of VMSAGE

VMSAGE is initialized inside a rack by configuring the logical quality in descending order.

In a typical Cloud data centre, turning on and shutting down VMs will cause a discrete distribution of VMs and thus, after a while, it will be difficult to implement precise temperature control. Therefore, we suggest to migrate VMs to realize the appropriate aggregation of VMs. In order to ensure the stability and QoS of the system, VMs need to be spread in a diffusing way from the nodes whose temperature or resource consumption is close to the upper limit. The ideal result of VMs migration should include a quick migration involving the minimum number of VMs with the minimum cost, resulting in the temperature or the load of the node to return back to an acceptable status. The goal of *VMSAGE* is to make sure that the maximum number of servers in the system enter the set of type C servers and also, to shut down the servers with low utilization.

The performance monitoring of servers should be conducted once after a time period t. Once the temperature is too high, or the utilization rate is too high or low, we must send the "migrate" signal to the corresponding machine. Once the number of "migrate" signals received by the monitoring unit exceeds the threshold value, we start the operation of *VMSAGE*. Then, VMs are gradually aggregated orderly and *VMSAGE* will terminate when **one** of the following conditions is true:

- All servers are of type C.
- All servers are of type B.
- All servers are of type A or C.
- All servers are of type B or C.
- All servers are of type D.

In order to further improve its performance and reduce the computational overhead, *VMSAGE* carries out the two-dimensional target selection, as follows:

> ***VMSAGE* Step 1:** Update the status of servers regularly. Once the demands are met for the algorithm to start, go to Step 2, otherwise maintain the present situation.

***VMSAGE* Step 2:** If a VM running on a type D server has no real task to execute, it will be written to the hard disk, then proceed with Step 3. Otherwise, the procedure jumps directly to Step 3.

***VMSAGE* Step 3:** Select the VM to be migrated at present and the corresponding target rack.

***VMSAGE* Step 4:** Select the target server and start the migration procedure. The workflow will not end unless the migration procedure is complete.

In practice, for the simplification of the workflow, *VMSAGE* must be further improved. For a server of type D, the number of VMs that must be migrated away is calculated. Next, the server's type will change to C, which can be used to define a new type of server, called A'; type A' server will be treated in the same way as type A server. We set a priority list to replace the assignment of the initial speed. During each iteration, the calculation of speed will result in computational overhead. Since the initial speed and the gravitation are established for an iteration, the priority list can be used to sort the virtual machines without calculating the speed for multiple times.

The iteration process for optimized rack selection is the following:

Iter Step 1: Get the list of servers of type D and type A. Calculate the logical quality of each rack and get all servers of type A' in type D. The specific procedure for separating servers of type A' in type D proceeds as follows:

(a) Sort all VMs in a type D server in descending order and write the corresponding number to a queue denoted by L.

(b) Current total resources of type D servers minus queue L column header serial number is corresponding to the virtual machine occupancy resources, calculate the type D server, and decide whether change to type C server; then, queue L column header serial number to write queue M.

(c) If the type D server is predicted to fall to type C, stop the calculations. The VMs that correspond to the numbers in queue M compose the type D servers, from which type A' servers are separated.

(d) If the type D server is not predicted to fall to type C, go back to Step 1b.

Iter Step 2: Sort all VMs on servers of type A' and type A in descending order, based on their logical quality; VMs on servers of type A' have higher priority than VMs on servers of type A.

Iter Step 3: Calculate the gravitation of each rack. The rack with the maximum gravitation is the target rack.

Iter Step 4: First, conduct parallel computing of servers of type A', and traverse all the servers in the queue, in order to determine the target rack of the first virtual machine in each servers queue. Then, conduct serial computing of servers of type A.

Iter Step 5: If VMs have not been fully migrated yet, go back to Step 4. Otherwise, end the algorithm.

The iterative procedure for selecting a server proceeds as follows:

Iter Step 1: Calculate the gravitation of servers of type B in the rack.

Iter Step 2: Calculate the modified gravitation and the direction of the resultant force.

Iter Step 3: In direction of joint force, the nearest server of type B is the target server. If there are more than one servers which satisfy the conditions, then select the server closest to the ground.

The monitoring operation mechanism with *VMSAGE* for the entire data centre is shown in Figure 5.

The computational cost of *VMSAGE* depends on the time complexity of the modified gravitation, which is $O(n^2)$. The logical qualities of servers should be

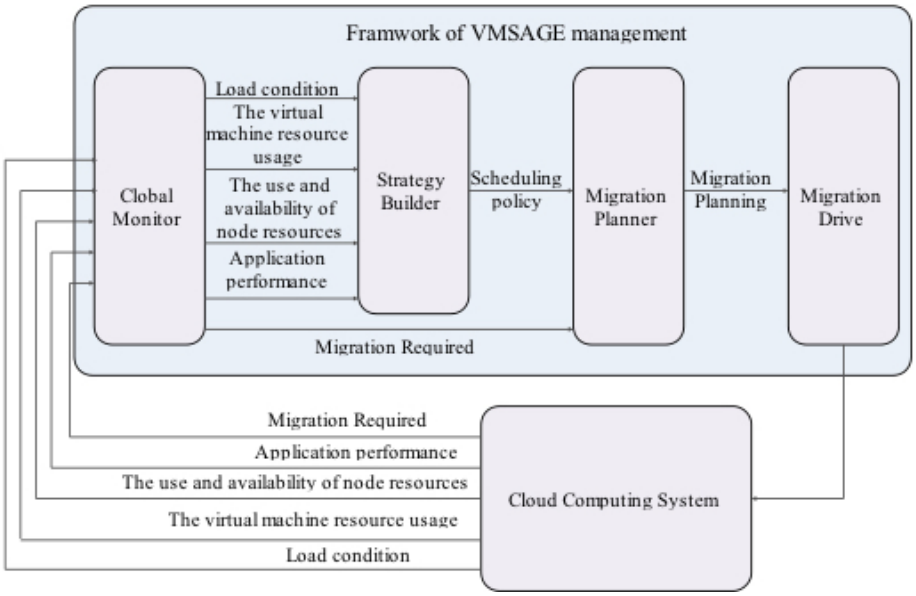


Figure 5: *The monitoring operation mechanism of* VMSAGE.

updated in real time and also be sorted. In this way, the correction of gravitation does not need to conduct traverse calculation, but only needs to seek the best solution from top to bottom. The idea is to use space to save time, and if the force table is used, the time complexity of algorithm will be reduced to $O(n)$.

5. Experiments and Performance Analysis

This section presents the experimental setup and the results of our evaluation.

5.1. Environment and performance index

We present an experimental analysis based on the CloudSim simulation framework [31]. The algorithm is evaluated and analyzed through the construction of the energy consumption performance of the data centre model. The experiments compare our solution with the Best Fit Heuristic (BFH) and the dynamic voltage and frequency scaling (DVFS). With regards to the configuration of the data centre, we set it to the air cooling mode, in which the underlying

server has the best cooling performance. The network topology adopts an hierarchical structure.

The formula used in this algorithm is highly adjustable according to different data centres. In order to integrate the CPU, RAM and other resources into the formula, the original unit is not used for each performance measurement. Instead, we use the percentage of the current CPU occupancy rate in the total CPU resources multiplied by the weight coefficient. In this way, the specific model of each part of the server does not affect the algorithm. To analyze the energy consumption of this experiment, we use the polynomial nonlinear model of [32]. By using the energy consumption model based on the system's usage, we can effectively estimate the impact of a single application on the server.

The energy consumption prediction model of a server is calculated using the following formula (k_i is the empirical coefficient, U_i is the utilization rate):

$$P = P_{static} + \sum_{i \in J} k_i U_i \qquad J \in \{CPU, RAM, DISK, I/O, ...\}$$

Furthermore, we set $\alpha(\nu_{jk}) = 0$, $\beta(t_i) = 0$, while t_i is calculated through prediction, not by using the actual monitoring value. The data centre uses Core i7 3770 processors, and according to the gray prediction, the fitting function used is equal to $t_1 = 28 \times e^{0.0014M(s_t)}$, while $G = 1$, $b = 2$, $L(s_i, \nu_{jk}) = 1$, and $g = 10$. In addition, the server uses the data measured by Dawning PHPC100 where $P_{static} = 180w$, $J = \{CPU\}$, $k_j = 1$.

Finally, the algorithm coefficients used in this paper are empirically calculated and their values should vary in other data centres with different configuration, in accordance with the corresponding circumstances.

5.2. Experimental Results

Our experimental analysis includes the following:

- Impact of the threshold values in *VMSAGE*.
- Energy consumption tracking in *VMSAGE*.
- Performance evaluation of *VMSAGE* versus DVFS and BFH.

- Thermal distribution tracking.

5.2.1. Impact of the threshold values on the VMSAGE effects

Let us assume that the data centre has 5 racks, each rack has $5 * 5$ servers running $125 * 5 = 625$ virtual machines, where each virtual machine is assigned the same amount of tasks. When the task is completed, the BFH algorithm is based on the energy consumption of 5.078 kw.h. Figure 6 shows Formula (5) when γ_1 is in the range from 0.2 to 0.4, γ_2 is from 0.4 to 0.6, and γ_3 is from 0.8 to 0.95. Due to the change in the threshold values of type A, B and C servers, we draw the Heat Map of energy conservation by using *VMSAGE* compared to BFH, in which, the numerical value of colour corresponds to the energy saved in the current threshold condition.

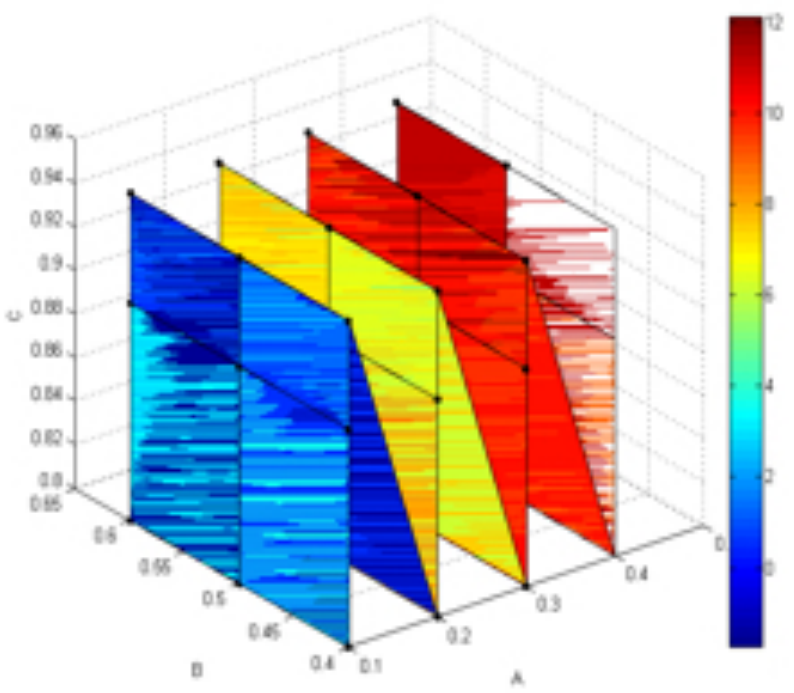


Figure 6: *Effect of server threshold on energy saving.*

The results indicate that when the number of servers is relatively small, the size of γ_1 has direct impact on the performance of *VMSAGE*. Under the same conditions, each virtual machine must complete the task a total of 5 times, and at this point, the energy consumption by BFH is 5.0468 kw.h. Figure 7 shows the performance of *VMSAGE* under different threshold values.

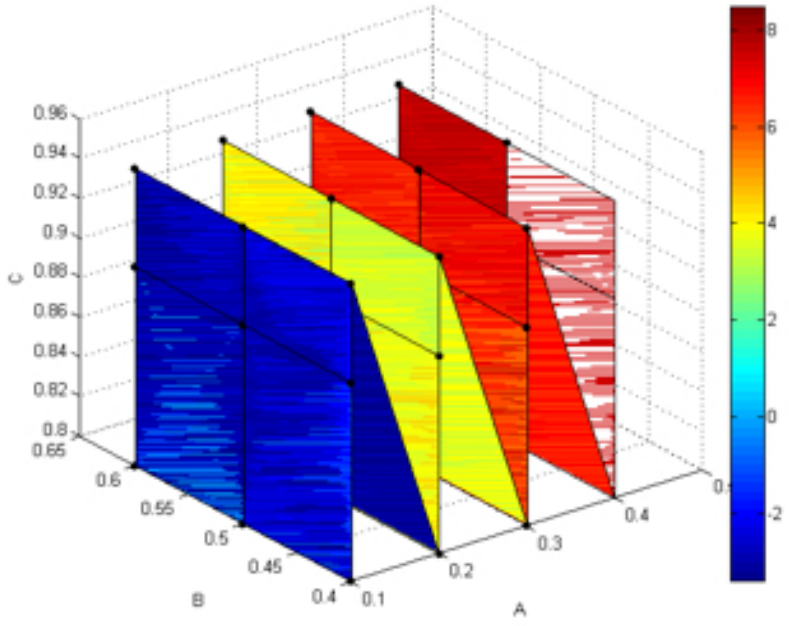


Figure 7: *Effect of server threshold on energy conservation with the increase of work amount.*

As we observe, the performance of the *VMSAGE* and BFH algorithms is not closely related to the task quantity. We proceed by selecting the two-experiment combination of γ_1, γ_2, γ_3 values with better performance, and conduct the *VMSAGE* energy consumption tests using 100 racks, where each rack has $10 * 10$ servers, resulting in a total of $5,000$ virtual machines. As we observe in Table 3, combination 8 has the best performance. As a result, the subsequent experiments use the threshold values of $\gamma_1 = 0.3$, $\gamma_2 = 0.5$, and $\gamma_3 = 0.95$ during calculations. In the actual environment, the threshold value should be

determined for the data centre first.

	γ_1	γ_2	γ_3	Energy Consumption
1	0.1	0.4	0.8	38.93
2	0.1	0.5	0.8	39.09
3	0.1	0.6	0.8	39.08
4	0.2	0.4	0.8	36.65
5	0.2	0.5	0.8	36.90
6	0.2	0.6	0.95	35.56
7	0.3	0.4	0.9	34.74
8	0.3	0.5	0.95	33.94
9	0.3	0.6	0.9	35.18
10	0.4	0.5	0.9	36.19
11	0.4	0.6	0.9	36.19

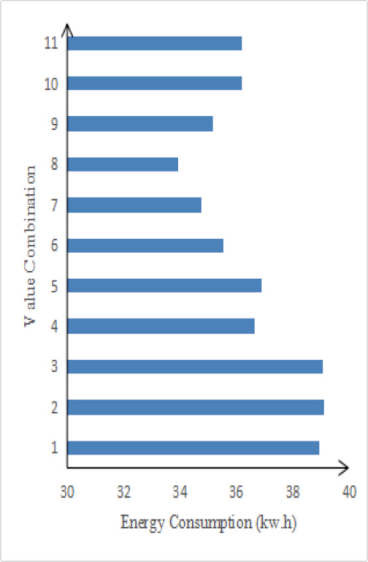


Table 3: *Find the threshold value with the best performance.*

5.2.2. Energy consumption tracking of VMSAGE server

We simulate the *VMSAGE* algorithm in an 8-server setup inside our data centre. In this process, hosts 1-4 are in the same rack, hosts 5-8 are in another rack, while hosts 1, 2, 5, and 6 are all underlying servers. As shown in Figure 8, since the underlying server has better cooling performance, so it can be seen that in the end, the top-level servers become dormant after completing the task, such as hosts 5-8.

5.2.3. Comparison among VMSAGE, DVFS, and BFH

For the next set of experiments, we set $\gamma_1 = 0.3$, $\gamma_2 = 0.5$, and $\gamma_3 = 0.95$, while the remaining conditions remain the same, in order to compare the energy consumption of each individual algorithm. The results shown in Figure 9 indicate that when the scale is further expanded, DVFS has the worst performance, while *VMSAGE* continuously has the best.

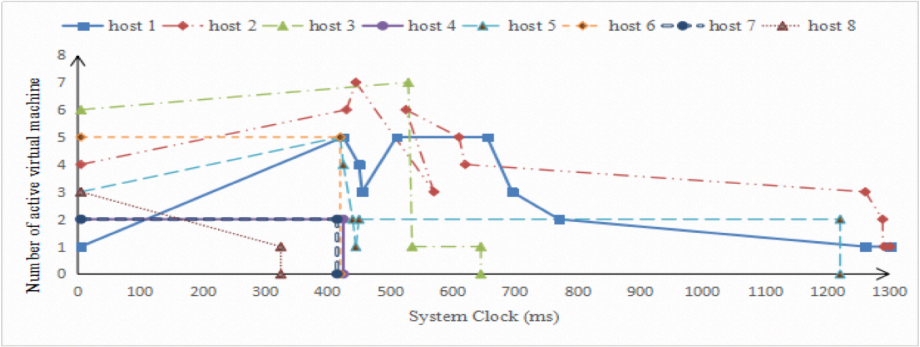


Figure 8: *Host energy tracking.*

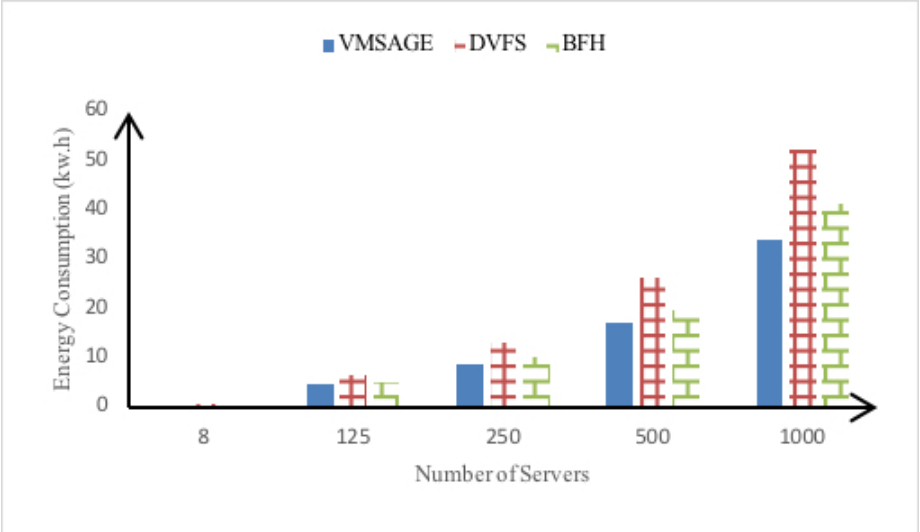


Figure 9: *Horizontal comparison of all algorithms.*

Figure 10 shows the ratio between the energy saved when using the BFH algorithm minus the energy saved when using the *VMSAGE* algorithm for different number of servers inside the data centre. According to the results, when the data centre is very small, consisting only of 8 servers, *VMSAGE* saves more energy than BFH. Also, Figure 10 shows that the average energy consumption of each server increases steadily, something which is true for the depicted ratio as well. Therefore, we conclude that *VMSAGE* is suitable to large-scale data centres. For the next experiment, we maintain the same experimental condi-

tions but this time, we increase the number of tasks to be executed. The energy consumption performance of all three algorithms is shown in Figure 11.

Figure 10: *Ratio between the energy saved when using the BFH algorithm minus the energy saved when using the* VMSAGE *algorithm and the current number of servers.*

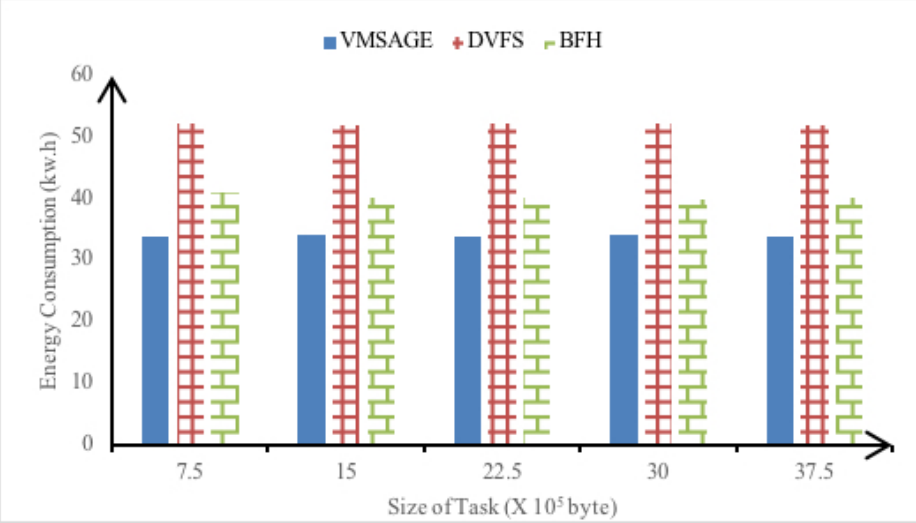


Figure 11: *Energy consumption performance of three algorithms under different task quantity.*

We observe that there is almost no relation between energy conservation and the number of tasks. Compared to BFH, the energy saved by *VMSAGE* is shown in Figure 12. We observe that there is no significant difference, which means that the main cause for difference in energy consumption is the number of virtual machines. However, this observation might be explained by the fact that the experimental environment is a virtual environment and that the tasks bound for each virtual machine is fixed. In reality, when every virtual machine is not allocated with the same number of tasks, we can convert this situation into various VMs that will share these tasks, which actually changes the problem into increasing the number of virtual machines. In this way, the state of the system will resemble the one used in our experiment and in this case, *VMSAGE* is still dominant.

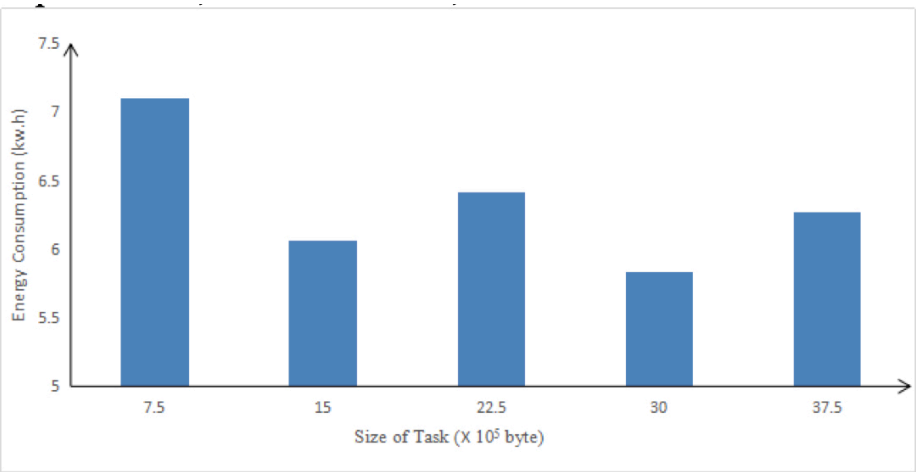


Figure 12: *Energy conservation of* VMSAGE *saving compared to BFH.*

For the $10 * 10 * 10$ data centre configuration, we launch $4,000$, $5,000$, $7,000$ and $8,000$ VMs respectively. The comparison of the three algorithms regarding their energy consumption is shown in Figure 13. We observe that the three algorithms are not sensitive to the number of VMs and thus, during the expansion of the data centre, the performance of the algorithms will remain stable.

Under the same conditions, the required server number might change. In

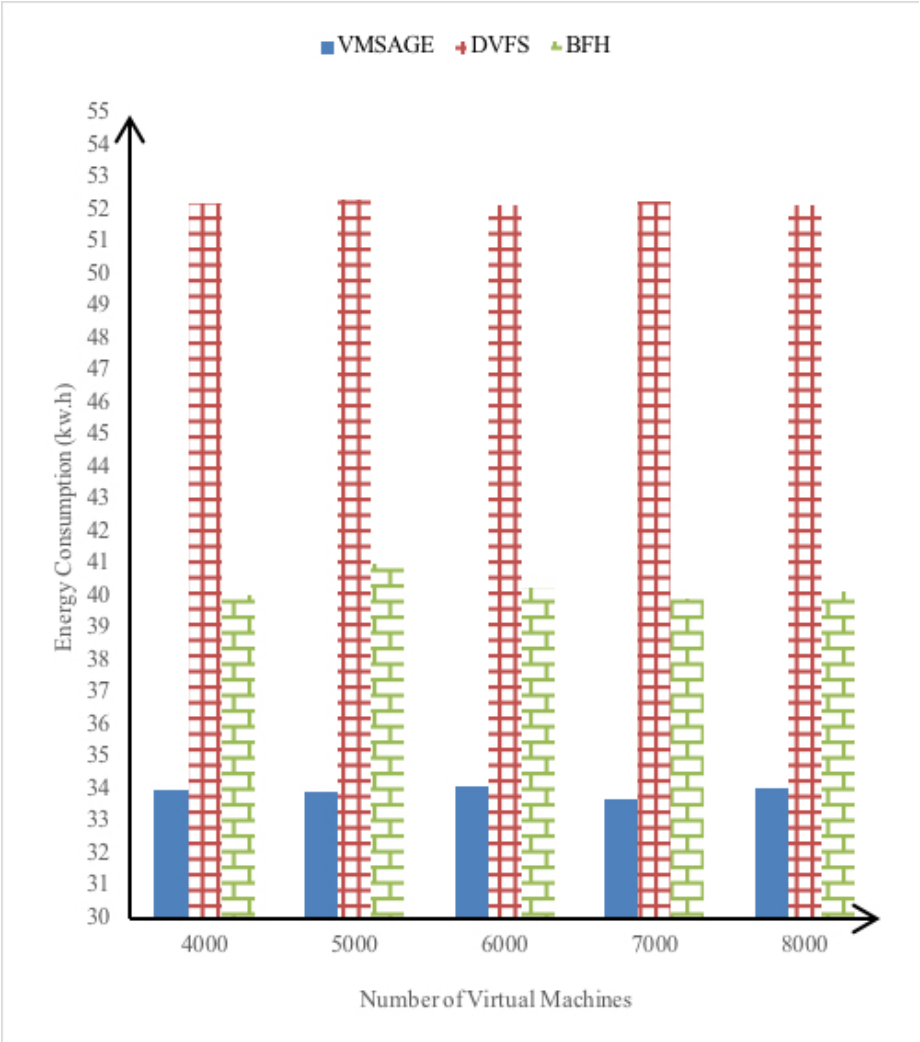


Figure 13: *Energy consumption of three algorithms with different number of virtual machines.*

order to complete the same amount of tasks, the number of virtual machines which need to be migrated when using the *VMSAGE* and BFH algorithms respectively is shown in Figure 14. The results indicate that the number of virtual machines that BFH requires is larger than those required by *VMSAGE* in the same situation. Therefore, when the data centre is larger, the number of VMs that must be migrated when the *VMSAGE* algorithm is used is less

compared to BFH's. That is to say, the greater the scale of the data centre, the better the performance of *VMSAGE*.

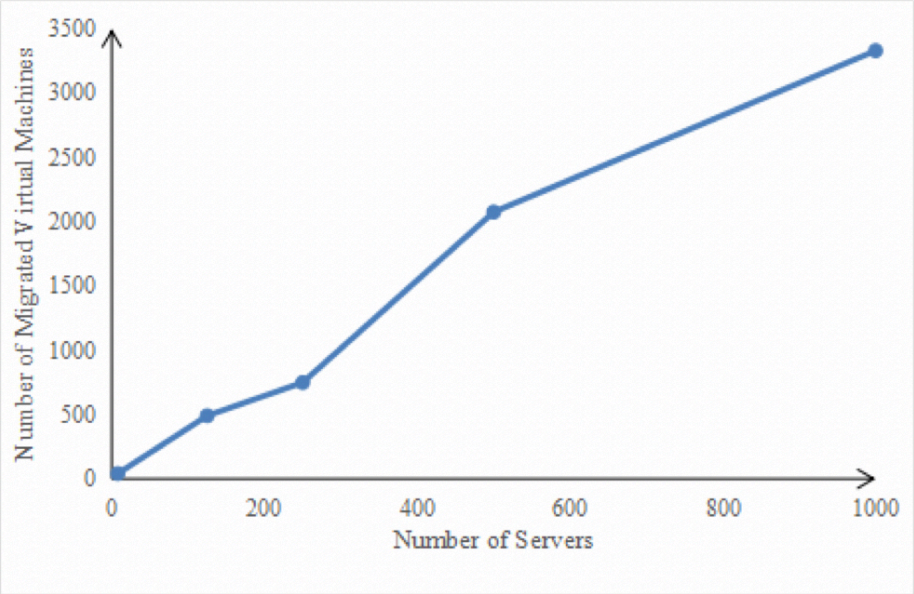
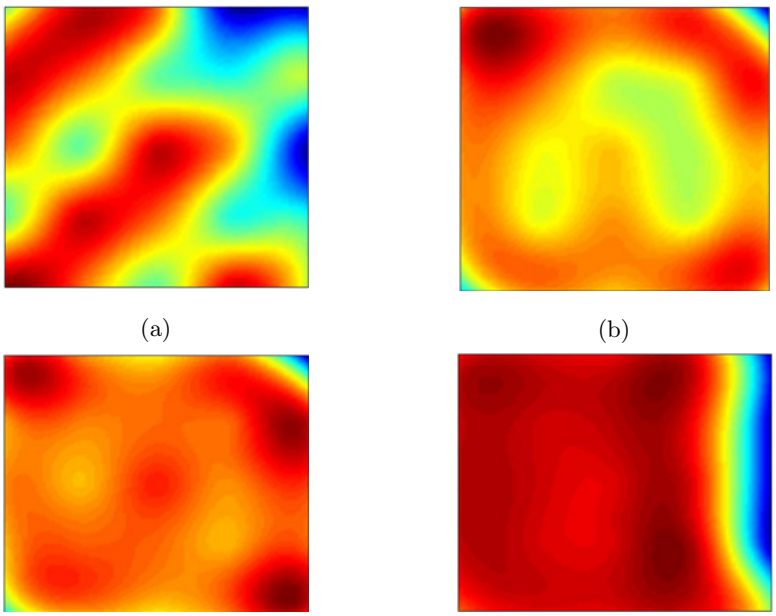


Figure 14: *The ratio between the number of migrated virtual machines with BFH algorithm and the number of virtual machines with VMSAGE.*

To continue with, it is difficult to accurately simulate the data of heat distribution, but the data can be estimated. A virtual machine is counted as a heat point to generate a top view graph of the computer rooms heat distribution. As shown in Figure 15a, because of the initial placement of the virtual machines, the room temperature experienced an uneven distribution. After a period of time, the room's temperature became gradually warmer, and finally, entered a temperature distribution scattered state, as shown in Figure 15b. At the same time, we make use of BFH and *VMSAGE* to migrate virtual machines. The heat map of the room's temperature distribution after a single treatment is shown in Figures 15c and 15d respectively. The migration strategy of BFH causes several temperature concentration points, while the temperature distribution of *VMSAGE* is even; some servers are even closed.



(a) (b)

(c) (d)

Figure 15: *A heat map of the computer room's heat distribution.*

5.2.4. Performance analysis

The experimental analysis shows that by using a proper VM migration scheduling algorithm, we can enhance the performance of the entire data centre and effectively save energy. The *VMSAGE* algorithm has significant optimization effects in improving the utilization rate of resources and reducing energy consumption and is also a feasible way to make decisions in real data centre environments. Also, the larger scale of the data centre, the better results will be achieved by the *VMSAGE* algorithm. This experiment uses the data centre with air cooling mode, so in theory, the closer to the bottom, the better cooling performance it has. However, during the construction of the actual data centre, the location with the best cooling effect may vary. Therefore, we suggest modifying the formula of energy consumption to adapt for data centres with various heat dissipation effects.

The algorithm solves the thrashing and time delay problems of VM migration. We define as "thrashing" of VMs the case where a VM which has just been migrated is transferred back, or if there are frequent migrations between various regions. The VM in each server is processed through the queue so the migrated VM will move into the end of queue to eliminate the thrashing problem. Furthermore, in our algorithm, except that the new VM can be placed at any time, other migration tasks are conducted in batch, and also the target server must be reachable, so it is not necessary to add the delay function. The algorithm is applicable to any data centre. The experiment proves that the algorithm can achieve good results in the calculation of the correlation coefficient. Also, because it avoids the overheating of server, it can significantly reduce the servers downtime caused by interruption of service. Finally, the SLA evaluation will also increase theoretically.

6. Conclusions

A Cloud computing system based on virtualization technology can enhance the flexibility of load deployment through dynamic resource expansion. In this

paper, we propose a new strategy of virtual machine management by combining the physical concepts with the actual situation inside Cloud data centres. Through experimental verification and after comparing our algorithm against two well-known approaches, we prove that under different data centre sizes, *VMSAGE* can save about 20% and 10% of energy consumption, when compared to DVFS and BFH respectively. What is more, *VMSAGE* compared to BFH can save about 20% of the number of virtual machines that must be migrated. The experimental analysis demonstrates that the algorithm is reasonable and feasible and that it can achieve good results regarding energy conservation.

As future work, we plan to incorporate network congestion into the design of our algorithm. In addition, we plan to consider deployment and migration of VMs in the global data centre.

References

[1] C. Wang, K. Ren, J. Wang, Secure optimization computation outsourcing in cloud computing: A case study of linear programming, IEEE transactions on computers 65 (1) (2016) 216–229.

[2] D. Yuan, X. Liu, Y. Yang, Dynamic on-the-fly minimum cost benchmarking for storing generated scientific datasets in the cloud, IEEE Transactions on Computers 64 (10) (2015) 2781–2795.

[3] X. Yin, Analysis of large data center air conditioning system energy saving and research on the method, Designing Techniques of Posts and Telecommunications (2015) 16–21, `http://dx.doi.org/10.16463/j.cnki.issn1007--3043.2015.01.004`.

[4] E. Wang, Fusion architecture leading future cloud computing data center, `http://net.zdnet.com.cn/network_security_zone/2016/0518/3077530.shtml`.

[5] B. Heller, S. Seetharaman, P. Mahadevan, Y. Yiakoumis, P. Sharma,

S. Banerjee, N. McKeown, Elastictree: Saving energy in data center networks., in: Nsdi, Vol. 10, 2010, pp. 249–264.

[6] P. Barham, B. Dragovic, K. Fraser, S. Hand, T. Harris, A. Ho, R. Neugebauer, I. Pratt, A. Warfield, Xen and the art of virtualization 37 (5) (2003) 164–177.

[7] M. D. De Assuncao, J.-P. Gelas, L. Lefevre, A.-C. Orgerie, The green grid5000: Instrumenting and using a grid with energy sensors, in: Remote Instrumentation for eScience and Related Aspects, Springer, 2012, pp. 25–42.

[8] Y. Zhao-Hui, J. Qin-Ming, Power management of virtualized cloud computing platform, Chinese Journal of Computers 6 (2012) 015.

[9] M. Yakhchi, S. M. Ghafari, S. Yakhchi, M. Fazeli, A. Patooghi, Proposing a load balancing method based on cuckoo optimization algorithm for energy management in cloud computing infrastructures, in: Modeling, Simulation, and Applied Optimization (ICMSAO), 2015 6th International Conference on, IEEE, 2015, pp. 1–5.

[10] S. Yakhchi, S. Ghafari, M. Yakhchi, M. Fazeli, A. Patooghy, Ica-mmt: a load balancing method in cloud computing environment, in: Web Applications and Networking (WSWAN), 2015 2nd World Symposium on, IEEE, 2015, pp. 1–7.

[11] C. Kachris, I. Tomkos, A survey on optical interconnects for data centers, IEEE Communications Surveys & Tutorials 14 (4) (2012) 1021–1036.

[12] S. Lai, The design of monitoring system and analyze local hot spot in it-rooms thermal environment, Master's thesis, Dept. Anhui University of Technology, Anhui, China (2014).

[13] L. A. Barroso, J. Clidaras, U. Hlzle, The Datacenter as a Computer: An Introduction to the Design of Warehouse-Scale Machines, Second Edition,

2013.
URL http://dx.doi.org/10.2200/S00516ED2V01Y201306CAC024

[14] Y. X. Zou Q, Hao Z, Live migration based on the characteristics of operation stages for virtual machine, Journal of Software (2016) 170–179.

[15] H. Chen, X. Zhu, H. Guo, J. Zhu, X. Qin, J. Wu, Towards energy-efficient scheduling for real-time tasks under uncertain cloud computing environment, Journal of Systems and Software 99 (2015) 20–35.

[16] A. Beloglazov, R. Buyya, Optimal online deterministic algorithms and adaptive heuristics for energy and performance efficient dynamic consolidation of virtual machines in cloud data centers, Concurrency and Computation: Practice and Experience 24 (13) (2012) 1397–1420.

[17] G. Zhu, K. Li, Y. Liao, Toward automatically deducing key device states for the live migration of virtual machines, in: Cloud Computing (CLOUD), 2015 IEEE 8th International Conference on, IEEE, 2015, pp. 1025–1028.

[18] W. Fang, X. Liang, S. Li, L. Chiaraviglio, N. Xiong, Vmplanner: Optimizing virtual machine placement and traffic flow routing to reduce network power costs in cloud data centers, Computer Networks 57 (1) (2013) 179–196.

[19] C.-M. Pintea, C. Pascan, M. Hajdu-Măcelaru, Comparing several heuristics for a packing problem, International Journal of Advanced Intelligence Paradigms 4 (3-4) (2012) 268–277.

[20] K. Le, R. Bianchini, J. Zhang, Y. Jaluria, J. Meng, T. D. Nguyen, Reducing electricity cost through virtual machine placement in high performance computing clouds, in: Proceedings of 2011 International Conference for High Performance Computing, Networking, Storage and Analysis, ACM, 2011, p. 22.

[21] F.-H. Tseng, C.-Y. Chen, L.-D. Chou, H.-C. Chao, J.-W. Niu, Service-oriented virtual machine placement optimization for green data center, Mobile Networks and Applications 20 (5) (2015) 556–566.

[22] M. Cardosa, M. R. Korupolu, A. Singh, Shares and utilities based power consolidation in virtualized server environments, in: Integrated Network Management, 2009. IM'09. IFIP/IEEE International Symposium on, IEEE, 2009, pp. 327–334.

[23] H. N. Van, F. D. Tran, J.-M. Menaud, SLA-aware virtual resource management for cloud infrastructures, in: Computer and Information Technology, 2009. CIT'09. Ninth IEEE International Conference on, Vol. 1, IEEE, 2009, pp. 357–362.

[24] H. Goudarzi, M. Ghasemazar, M. Pedram, SLA-based optimization of power and migration cost in cloud computing, in: Proceedings of the 2012 12th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (ccgrid 2012), IEEE Computer Society, 2012, pp. 172–179.

[25] H. Goudarzi, M. Pedram, Energy-efficient virtual machine replication and placement in a cloud computing system, in: Cloud Computing (CLOUD), 2012 IEEE 5th International Conference on, IEEE, 2012, pp. 750–757.

[26] C. Dupont, T. Schulze, G. Giuliani, A. Somov, F. Hermenier, An energy aware framework for virtual machine placement in cloud federated data centres, in: Future Energy Systems: Where Energy, Computing and Communication Meet (e-Energy), 2012 Third International Conference on, IEEE, 2012, pp. 1–10.

[27] B. Li, J. Li, J. Huai, T. Wo, Q. Li, L. Zhong, Enacloud: An energy-saving application live placement approach for cloud computing environments, in: Cloud Computing, 2009. CLOUD'09. IEEE International Conference on, IEEE, 2009, pp. 17–24.

[28] A. Baruchi, E. T. Midorikawa, L. M. Sato, Reducing virtual machine live migration overhead via workload analysis, IEEE Latin America Transactions 13 (4) (2015) 1178–1186.

[29] S. Vahora, R. Patel, H. Patel, S. Patel, Efficient virtual machine management based on dynamic workload in cloud computing environment, in: Advance Computing Conference (IACC), 2015 IEEE International, IEEE, 2015, pp. 603–608.

[30] C. Yan-guang, L. Ji-sheng, Derivation and generalization of the urban gravitational model using fractal idea with an application to the spatial cross-correlation between beijing and tianjin, Geographical Research 21 (6) (2002) 742–752.

[31] R. N. Calheiros, R. Ranjan, A. Beloglazov, C. A. De Rose, R. Buyya, CloudSim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms, Software: Practice and experience 41 (1) (2011) 23–50.

[32] Z. F. Luo L, Wu W, Energy modeling based on cloud data center, Journal of Software (2014) 1371–1387, http://dx.doi.org/10.13328/j.cnki.jos.004604.

[33] T. Horvath, T. Abdelzaher, K. Skadron, Dynamic Voltage Scaling in Multitier Web Servers with End-to-End Delay Control, IEEE Transactions on Computers 56.4, 2007, pp. 444–458.

[34] T. Mofolo, R. Suchithra, Heuristic based resource allocation using virtual machine migration: a cloud computing perspective, International Refereed Journal of Engineering and Science, vol. 2, no. 5, 2013, pp. 40-45.

[35] C. W. Kurnik, R. Huang, E. Masanet, Chapter 20: Data Center IT Efficiency Measures Evaluation Protocol. The Uniform Methods Project: Methods for Determining Energy Efficiency Savings for Specific Measures, 2017, http://dx.doi.org/10.2172/1408088.